REMARKS

Applicant amended independent claim 13 for greater clarity in accordance with the examiner's comments as specified in the Final Action of August 15, 2006. After this amendment, claims 1-15 are pending in the above-identified patent application. Claims 1, 7, 10 and 13 are independent claims.

The examiner rejected claim 1 under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,401,155 to Saville. The examiner rejected claims 7-15 under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,205,468 to Diepstraten et al. The examiner also rejected claims 2-6 under 35 U.S.C. §103(a) as being unpatentable over Saville in view of U.S. Patent No. 5,815,698 to Holmann.

With respect to applicant's independent claim 1, the examiner stated in the October 3, 2006, Advisory Action:

> Applicant argues the rejection of claim 1 on page 8 of the after-final remarks, in substance that:
>
> "Applicant's independent claim 1 recites "a context branch instruction that, when executed causes a data processing apparatus to: cause an instruction stream to branch to another instruction of the instruction stream associated with a label specified by the context branch instruction based on whether or not a current context number matches a context number specified by the context branch instruction."
> Thus, applicant's context branch instruction performs the branching operation on the basis of the evaluation of whether the current context number matches the context number specified in the context branch instruction. That evaluation determines whether the instruction will or will not perform the branching operation...Seville's "SWITCH TO PO" instruction causes the processor to unconditionally switch from the current processor to the context whose address in memory is indicated by the content of PO. The "SWITCH TO PO" instruction, however, is not a branching operation that causes a branch to a different instruction in the instruction stream if the condition of the branch is fulfilled, as recited in applicant's claim 1."
>
> While fully considered, this argument has been found non-persuasive by the examiner because applicant is reading the claim too narrowly. Due to the or-clause in the second to last line of the claim, the claim effectively reads as "causing an instruction stream to branch to another instruction of the instruction stream...based on a current context number matching a context number specified by the branch instruction or based on a current context number not matching a context number specified by the branch instruction." If a claim states "A or B" and the prior art only teaches B, then the prior art still anticipates the claim. In this case, the examiner is only concerned with the non-matching portion. That is, Saville has taught

> "causing an instruction stream to branch to another instruction of the instruction stream...based on a current context number not matching a context number specified by the branch instruction." The reason this is taught is because the specified context number is not the same as the current context number, a branch (flow disruption) occurs in the program code being executed. Regardless if it is an unconditional branch or not, the branch still occurs because the current number does not match the specified number, i.e, the branching is based on wanting to go to as different context.

Preliminarily, applicant notes that independent claims 7, 10 and 13 recite "evaluating a context number of an executing context to determine whether the context number of the executing context matches a context number specified by a context branch instruction; and branching to a specified instruction in accordance with evaluating the context number of the executing context," or similar language. The language recited in independent claims 7, 10 and 13 does not include the use of the wording "or not", and thus the examiner's comments in the advisory action do not apply to the language recited in applicant's claims 7, 10 and 13.

As for applicant's independent claim 1, applicant amended independent claim 1 to include language similar to that appearing in independent claims 7, 10 and 13, and claim 1 now recites that the branch to another instruction in an instruction stream is based on an evaluation of whether a current context number matches the context number specified in a context branch instruction. Additionally, applicant removed the wording "or not," previously appearing in independent claim 1, for greater clarity.

Applicant's independent claim 1 recites "a context branch instruction that, when executed causes a data processing apparatus to: cause an instruction stream to branch to another instruction of the instruction stream associated with a label specified by the context branch instruction based on an evaluation of whether a current context number matches a context number specified by the context branch instruction." Thus, applicant's context branch instruction performs an evaluation of whether the context number matches the context number specified in the branch instruction, and performs a branch operation based on that evaluation.

In contrast, Saville describes multiple-thread processing by the use of contexts (col. 1, lines 9-10). A processor 10 is used to control the multiple thread operation of Saville's system. Saville explains:

> **At time t_3, while running thread A, the processor reads the instruction SWITCH TO P0 TC from a respective location in the memory 20' identified by the program counter register PC. This instruction is provided to enable a switch from one thread to another which may not be identified by any of the pre-stored thread context pointers Ptr TC EXT 1, Ptr TC EXT 2 or Ptr TC EXT 3 in the memory 20'. To facilitate this switch, one of the instructions which precedes the SWITCH TO P0 TC instruction will pre-store in the P0 register, in the context register set, the address identifying the location of the context for the thread to be switched to. In this example, the switch will be from current thread A to a thread D located somewhere in the memory 20'. In execution of this instruction the processor:**
>
> **reads THREAD CONTEXT A into the memory location identified by the current thread context address contained in register PTR TC in the context register set 14;**
>
> **reads into the register PREV PTR TC the address currently contained in the register PTR TC;**
>
> **reads the address from the P0 register into the pointer register 122;**
>
> **reads the thread context (THREAD CONTEXT D) from the location in the memory 20' identified by the pointer register 122 into the context register set 14;**
>
> **begins running thread D. (Saville, col. 5, lines 25-51)**

Saville's "SWITCH TO P0" instruction causes the processor to unconditionally switch from the current context to the context whose address in memory is indicated by the content of P0. Accordingly, Saville's "SWITCH TO P0" instruction does not cause any type of evaluation to be performed, let alone an evaluation as to whether the current context number matches a context number specified in an instruction. Therefore, performance of the "SWITCH TO P0" instruction, or for that matter any of Saville's instructions, is not based on "an evaluation of whether a current context number matches a context number specified by the context branch instruction," as required by applicant's independent claim 1.

Accordingly, applicant's independent claim 1 is patentable over Saville.

Claims 2-6 depend from independent claim 1, and are therefore patentable for at least the same reasons as independent claim 1.

As noted, the examiner rejected claim 7 in the August 15, 2006, Final Action, as being anticipated by Diepstraten (as further noted, the examiner did not make reference to claims 7, 10

and 13 in the October 3, 2006, Advisory Action.) Specifically, the examiner stated in the final action:

> 13. Referring to claim 7, Diepstraten has taught a method of operating a processor comprising evaluating a context number of an executing context to determine whether the context number of the executing context matches a context number specified by a context branch instruction, and branching to a specified instruction in accordance with evaluating the context number of the executing context. See column 19, lines 40-55, and note that a current context number is compared to a new context number, which is inherently specified by an instruction as instructions cause action to occur in a system. The comparison is done to see if branching to a new context should occur.

Applicant respectfully disagrees with the examiner's contentions.

Independent claim 7 recites "evaluating a context number of an executing context to determine whether the context number of the executing context matches a context number specified by a context branch instruction; and branching to a specified instruction in accordance with evaluating the context number of the executing context."

Diepstraten describes a context controller for managing multitasking in a processor (Abstract). The context controller includes an event synchronizer 150 and an event prioritizer 152 (FIG. 5 and col. 17, lines 39-57). In relation to the event prioritizer 152, Diepstraten explains:

> **FIG. 7D defines operation during the second quarter of each cycle [of the event prioritizer process], a Qr-to Mf period, (a period from a quadrature clock rising edge Qr to the next master clock falling edge Mf). This is the time period when events are prioritized and the context switching decisions are made. The first set of actions (symbols 422-428) searches for a possible preemption. The search is depicted as an iterative process for clarity regarding the operation being performed. This operation is typically performed for all contexts in parallel. If the running context is in foreground, the search is over the range 0:ctx, whereas if the running context is in background the search is over the range 0:7 because all foreground contexts have priority over any background context (symbol 423). The priority encoding (symbol 424) is implicit in the ascending context number 424 (descending priority) sequence. If an active, foreground context is found, its number is recorded in nctx (symbol 452). Otherwise, a search (symbols 430-434) is conducted for an active background context starting at the indicated current background context and continuing to higher context numbers (modulo 8).**
>
> **If a time slice (the symbol 334 of FIG. 7B) ends at the master clock rising edge Mr 292 of this cycle, the indicated curBg will already have been incremented, meaning the search will start from the context after the one**

that is currently running and will only re-select the same context if no other contexts are in the queued state Qb. In the case of a preempted context in the background running state Rb that can now be resumed, this test (symbol 430) will exit immediately to a set new context number (nctx) 450. If either a foreground (set new context number (nctx) 452) or a background (symbol 450) search finds a context to run, the new context number (nctx) is compared with the current context number (ctx) (symbol 454) to determine whether a context switch is needed. If a context switch is not needed, no further context control activities occur during this cycle and the controller returns to a running state 458. (col. 19, lines 20-55)

Thus, Diepstraten's event prioritizer process is directed to context switching (i.e., switching from executing one context to executing a different context) not to branching to a different instruction of an instruction stream as in claim 7. Specifically, Diepstraten's event prioritizer process searches for contexts that need to be run. If during such a search a context is found, that context number is compared to the context number of the currently executing context to ensure that they are not the same (if they are, there is no need to perform any context switching). Diepstraten does not describe that either the context number identified through the search, or the currently running context number, is specified in a branch instruction or in any of Diepstraten's other instructions.

Accordingly, Diepstraten does not disclose or suggest at least the features of "evaluating a context number of an executing context to determine whether the context number of the executing context matches a context number specified by a context branch instruction; and branching to a specified instruction in accordance with evaluating the context number of the executing context," as required by applicant's independent claim 7. Applicant's independent claim 7 is therefore patentable over the cited art.

Claims 8-9 depend from independent claim 7 and are therefore patentable for at least the same reasons as independent claim 7.

Independent claims 10 and 13 recite "evaluate a context number of an executing context to determine whether the context number of the executing context matches a context number specified by the branch instruction; and branch to a specified instruction in accordance with evaluating the context number of the executing context," or similar language. For at least similar reasons as those provided with respect to independent claim 7, at least these features are not

disclosed by the cited art. Applicant's independent claims 10 and 13 are therefore patentable over the cited art.

Claims 11-12 depend from independent claim 10 and are therefore patentable for at least the same reasons as independent claim 10. Claims 14-15 depend from independent claim 13 and are therefore patentable for at least the same reasons as independent claim 13.

All of the dependent claims are patentable for at least the reasons for which the claims on which they depend are patentable.

In view of the foregoing, applicant respectfully submits that the application is in condition for allowance and such action is respectfully requested at the examiner's earliest convenience.

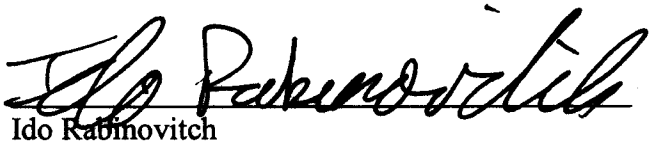
Canceled claims, if any, have been canceled without prejudice or disclaimer.

Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

No fee is believed due. Please apply any other charges to deposit account 06-1050, referencing attorney docket 10559-307US1.

Respectfully submitted,

Date: Nov. 14, 2006



Ido Rabinovitch
Attorney for Intel Corporation
Reg. No. L0080

Customer No. 20985
Fish & Richardson P.C.
Telephone: (617) 542-5070
Facsimile: (617) 542-8906